Laboratorio 5: "Tarjeta de Desarrollo Spartan-3"

Objetivos:

- Aprender el uso de dispositivos controlados por una FPGA en una tarjeta de desarrollo.
- Conocer la interfaz PS/2, el protocolo de comunicación y *scan codes* para la conexión de un teclado.
- Aprender a manejar pulsadores, interruptores, leds y display de 7 segmentos.
- Utilizar programación modular, para la reutilización de código en el diseño de circuitos complejos.
- Desarrollar e implementar métodos de prueba para módulos individuales y el circuito completo.

1 Recursos Disponibles

1.1 Tarjeta de Desarrollo Spartan-3 de Digilent

En este laboratorio se utilizarán algunos recursos internos que dispone la tarjeta de desarrollo con el fin de aprender su funcionamiento. Específicamente se utilizará la interfaz PS/2 para una conexión a teclado, y pulsadores e interruptores todos ellos para entrada de datos, para salida de datos se usarán leds y displays de 7 segmentos. Para cada uno de estos recursos se diseñarán pequeños módulos que permitan aprender su uso.

La interfaz PS/2 que dispone la tarjeta corresponde al estándar IBM y será presentada con más detalle en la sección 1.2. En la página del curso se ha dispuesto de material de apoyo adicional.

Tanto los pulsadores como los interruptores de la tarjeta están conectados cada uno a un pin de entrada de la FPGA y corresponden a un 1 ó 0 lógico dependiendo de su estado. Un detalle de la correspondencia de pines se encuentra en el manual del usuario, capítulo 4.

Además, la tarjeta dispone de 8 leds y 4 displays de 7 segmentos como salida de datos. Los leds se activan con un 1 lógico. Los segmentos de los displays se activan con un 0 lógico. Para reducir la cantidad de pines necesarios para la activación de los displays de 7 segmentos, éstos se encuentran conectados en paralelo. La activación de un display se logra mediante un pin de control conectado al ánodo común del display (mediante un 0 lógico conectado a la base de un transistor que alimenta al display). Con esto se logra reducir la cantidad de salidas requeridas de 36 a 12. Para poder observar varios dígitos encendidos será necesario multiplexar en el tiempo las señales de control, refrescando continuamente los displays. Una descripción detallada se encuentra en el capítulo 3 del manual del usuario.

1.2 Interfaz PS/2

La interfaz de comunicación PS/2 es una interfaz propuesta por IBM para comunicar dispositivos seriales en forma sincrónica, tanto teclado como Mouse. En la actualidad y antes de la masificación de la interfaz USB, la mayoría de los teclados correspondían a este tipo con las siguientes características:

- Gran número de teclas (101 a 104).
- Conector de 5 o 6 pines (incluyen adaptadores).
- Protocolo de comunicación serial bidireccional (PS/2).
- Garantizan sólo el conjunto 2 de los *scan codes*.
- Responden a todos los comandos enviados, sin embargo no actúan en todos ellos.

La interfaz física que usualmente se utiliza se muestra en la Figura 1.

La alimentación del teclado/mouse corresponde a Vcc = +5 [V] y una corriente máxima de 100 [mA]. Se recomienda no conectar el teclado a la tarjeta mientras se encuentre encendida. Las líneas de Data y Clock son ambas de colector abierto, con resistencias de "pull-up" para fijar ambas líneas en alto.

El mouse y teclado PS/2 implementan un protocolo de comunicación serial bidireccional. El bus de comunicación se encuentra en estado "*idle*" cuando ambas líneas (data y clock) se encuentran en alto. Este es el único estado en que al dispositivo (teclado o mouse) le está permitido enviar información al *host* (computador o en nuestro caso tarjeta de desarrollo Spartran 3). El *host* tiene el control último sobre el bus y puede inhibir la comunicación en cualquier instante, colocando la línea de clock en nivel bajo.

Plug

Socket

6-pin Mini-DIN (PS/2):
1 - Data
2 - No Implementado
3 - Ground
4 - Vcc (+5V)
5 - Clock
6 - No Implementado

Figura 1: Interfaz PS/2

El dispositivo siempre genera la señal de clock. Si el *host* desea enviar datos, debe primero inhibir la comunicación desde el dispositivo, colocando la línea de clock en nivel bajo. Luego debe colocar en nivel bajo la línea de data y subir la línea de clock. Este estado es conocido como "*Request to Send*", con lo cual el *host* señaliza al dispositivo que comience a generar pulsos de reloj a través de la línea clock para enviar los datos. Por lo tanto el bus puede estar en uno de los siguientes tres estados:

- **IDLE**: Data y Clock en alto
- **INHIBIT**: Data en alto y Clock en bajo
- **REQUEST TO SEND**: Data en bajo y Clock en alto

Todos los datos son enviados de un byte por vez y cada byte es enviado dentro de un frame de 11 o 12 bit:

- 1 bit de partida: siempre es 0.
- 8 bit de datos: se comienza por el menos significativo.
- 1 bit de paridad: se utiliza paridad impar.
- 1 bit de parada: siempre es 1.
- 1 bit de confirmación: sólo para comunicación desde el *host* al dispositivo.

El bit de paridad es colocado en alto si hay un número par de 1's en los bit de datos, y colocado en bajo si hay un número impar. La idea es que el número de 1's de los bit de datos más el bit de paridad siempre sean un número impar (paridad impar). Esto se utiliza para la detección de errores en la transmisión. Si el dispositivo detecta un error responde como que se le ha enviado un comando inválido. La Figura 2 muestra la comunicación del dispositivo con el *host*.

Cuando el teclado o mouse desean mandar información deben asegurarse que la línea de clock debe estar en alto por lo menos 50 [us] antes de que el dispositivo pueda comenzar a enviar datos. El teclado/mouse escribe un bit en la línea de data cuando el reloj está alto, y es leído en el *host* cuando el reloj está bajo. Como se trata de una comunicación desde el dispositivo hacia el *host* no se envía un bit de confirmación.

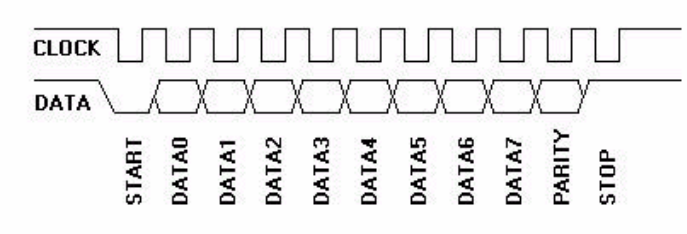

Figura 2: Comunicación PS/2 - *Host*

1.2.1 Scan Codes

En los teclados se distinguen distintos conjuntos de *scan codes* para identificar la tecla o teclas que se presionan. Se utiliza por defecto el conjunto 2. Cada tecla tiene asociado un *scan code* compuesto de dos códigos: un *make code*, que se emite cada vez que se presiona una tecla, y un *break code*, que se emite cuando se suelta la tecla. En general estos códigos suelen tener entre 1 y 2 byte, pero existen ciertos *scan codes* que son más largos debido a que corresponden a una combinación de teclas más elaborada.

En http://www.computer-engineering.org/ps2keyboard/scancodes2.html se encuentra disponible los *scan codes* del conjunto 2, así como también en el manual de referencia de la tarjeta de desarrollo capítulo 6, disponible en el sitio de la asignatura.

1.3 Driver PS/2

En el sitio de la asignatura se encuentra disponible un driver básico PS/2 escrito en lenguaje Verilog (`kbd.v`). La Figura 3 muestra la idea principal del funcionamiento del driver.

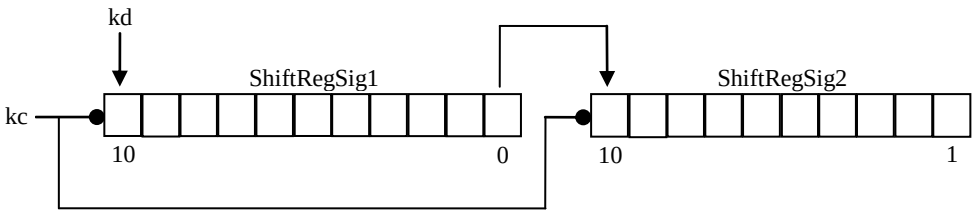

Figura 3: Conexión Lógica para la Captura de Datos en Serie

El driver de teclado realiza la captura de datos en serie vía la línea `kd`, para lo que utiliza 2 registros de desplazamiento de 11 y 10 bit respectivamente. El reloj de sincronismo `kc` proviene de la misma fuente de los datos y está conectado a ambos registros. El bit más significativo del registro 1 es el que recibe los datos seriales. Al capturar el valor hexadecimal "`F0`" en el registro 2, se ha recibido el *break code* de una tecla, cuyo valor está almacenado en el registro 1 entre los bit 8 al 1.

2 Actividades Previas

2.1 Contador BCD

En el sitio de la asignatura se encuentra el código en Verilog de un conversor binario a 7 segmentos (ssdec.v). Este código tiene como entradas el dato en binario de 4 bit, el estado de punto decimal y el tipo de dispositivo (ánodo o cátodo común). La salida del módulo es el estado de cada segmento del display, incluido el punto decimal. Debido a la configuración de los display 7 segmentos en la tarjeta de desarrollo, para poder observar varios dígitos encendidos será necesario multiplexar en el tiempo las señales de control, refrescando continuamente el display. Complete el módulo `display()` **respetando** las siguientes características de entrada y salida:

```
module display(clk, num, type, pts, ssg, dctl);
input clk, type;
input [3:0] pts;
input [15:0] num;
output [7:0] ssg;
output reg [3:0] dctl;

   // fill in

endmodule
```

donde `clk` es una señal de reloj con una frecuencia tal que permita multiplexar el control del display en el tiempo, `num` es el valor de entrada de 16 bit, `type` es el tipo de display (ánodo o cátodo común), `pts` es un *nibble* para indicar el estado del punto de cada display, `ssg` y `dctl`

corresponden a las salidas de los segmentos, incluido el punto, y las señales de control para el encendido del display.

Un contador BCD facilita la visualización decimal de cuentas binarias. El siguiente módulo es un contador en BCD de 4 dígitos:

```
module bcdcounter(rst, clk, value);
input rst, clk;                 // reset, clock
output reg [15:0] value;        // 4 bcd digits

   always @(negedge clk or posedge rst) begin
      if (rst == 1)
         value <= 0;
      else begin
         if (value[3:0] == 9) begin
            value[3:0] <= 0;
            if (value[7:4] == 9) begin
               value[7:4] <= 0;
               if (value[11:8] == 9) begin
                  value[11:8] <= 0;
                  if (value[15:12] == 9) value[15:12] <= 0;
                  else value[15:12] <= value[15:12] + 1;
               end else value[11:8] <= value[11:8] + 1;
            end else value[7:4] <= value[7:4] + 1;
         end else value <= value + 1;
      end
   end

endmodule
```

Utilizando los módulos anteriores, diseñe un módulo que permita la visualización en el display del tiempo transcurrido desde que se inicializó la FPGA en formato MM.SS. Construya un módulo de prueba para simular el diseño, incluyendo las secciones de inicialización que sean necesarias (pero que no son relevantes para la síntesis).

2.2 Uso de Pulsadores, Interruptores y Leds

Diseñe un módulo que permita asociar a cada interruptor de la tarjeta de desarrollo un led para desplegar el estado del interruptor respectivo. Añada la posibilidad de encender, apagar e invertir el estado de todos los leds al presionar el pulsador 0, 1 ó 2.

Construya un módulo de prueba para verificar el diseño mediante simulación.

2.3 Interfaz PS/2

Utilizando el módulo `display()` desarrollado en el punto 2.1, y el driver descrito en la sección 1.2, diseñe un módulo que permita capturar el *scan code* de la tecla presionada en los dígitos menos significativos del display. Si la tecla presionada es un número, desplegar en los dos dígitos más significativos su valor decimal (e.g. 01, 02), y "FF" si no lo es. Si es necesario, construya módulos de prueba para verificar su diseño. ¿Qué *scan code* obtendría para las teclas de flechas?

3 En el Laboratorio.

3.0 Revisión Actividades Previas

Muestre al profesor o ayudantes la simulación funcional y temporal de los módulos desarrollados en los puntos 2.1 y 2.2. Muestre al profesor el diseño del módulo del punto 2.3 que usa el driver PS/2.

3.1 Contador BCD

Sintetice el diseño del reloj BCD del punto 2.1 y verifique su funcionamiento.

3.2 Uso de Pulsadores, Interruptores y Leds

Sintetice el diseño del punto 2.2 y verifique su funcionamiento.

3.3 Interfaz PS/2

Sintetice el diseño del punto 2.3 y verifique su funcionamiento.

3.4 Máquina Digital

Use los interruptores para especificar una operación. Diseñe un circuito que permita realizar las siguientes operaciones:

Operación	Contenido del display y leds
0	Contador BCD ascendente **X** Hz.
1	Contador BCD descendente **Y** Hz.
2	Contador octal ascendente de X Hz.
3	Contador octal descendente de **Y** Hz.
4	Contador del número de teclas presionadas.
5	Cuenta el número de veces que se presiona la misma tecla.
6	2MSD del display muestran op. 0, 2LSD del display muestran op. 1.
7	Los puntos del display pestañean a X Hz, el display mantiene última operación.
8	Display on (todos los segmentos y puntos), se mantienen las cuentas.
9	Proponga un comportamiento dinámico para los leds y puntos del display.
10	2LSD del display indican el último pulsador accionado; se mantienen las cuentas.
11	Muestra el tiempo transcurrido entre 2 teclas presionadas en [ms] con máximo 9999.
12	Muestra el tiempo transcurrido según el punto 2.1.

El pulsador 0 hace un reset de todas las cuentas. El pulsador 1 muestra una rotación de los dígitos del display a la derecha en 1 posición, y el pulsador 2 lo mismo pero hacia la izquierda. El pulsador 3 hace que el display pestañee a Y Hz, manteniendo cuentas y respetando las operaciones. Los leds muestran en todo momento, salvo en la operación 9, la cuenta binaria de los 2 dígitos más significativos del display. Los valores de X, Y le serán asignados por el profesor o ayudante.